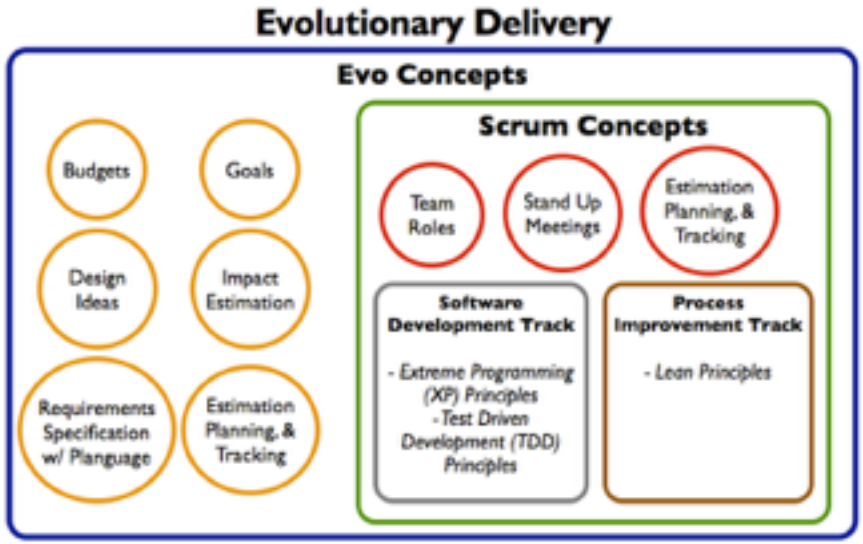


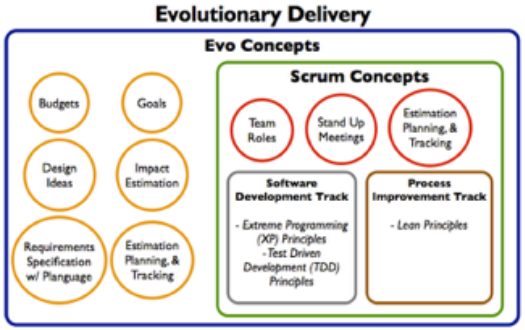
"Value delivery for Agile environments"
(Agile Methods Lack Result Management)

By Tom Gilb
MASTER 2016

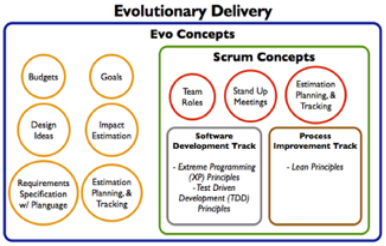
Summary
"Give Value, not Code"

- Conventional Agile methods (Scrum etc.) are fine for organising the programming tasks.
- But, they need to supplemented by an Agile Envelope

  - 'Evo' Method
  - Which focuses on
    - Delivery of useful results to stakeholders



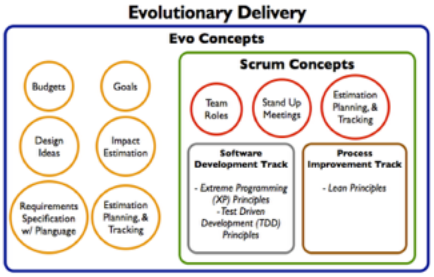

- In both Norway and USA we have recent experience from this combination (Evo+Scrum)
- Are you ready for the next step of Agile Matrurity?



Agile is an *improvement* but it's not *enough*

- Yes they work –
  - Agile methods (XP, Scrum) have proven themselves adept at delivering results quickly and agile is becoming more mature and accepted in the industry
- But where's the **alignment with business value**?
- Popular agile methods such as XP and Scrum **don't provide guidance**
  - on ensuring the agile team is implementing solutions
  - with the "biggest bang for the buck"
  - and make sure that business is getting the best ***value for their money***!
- **Alignment to Measurable Goals**
- In order for agile methods to transform, not only software projects, but also the way businesses **implement change** across their organization,
  - teams using agile methods must align their work with **higher-level business goals** and
  - measure their results, with **respect to helping organizations achieve their goals**!

"Just because you're Agile doesn't mean you're making Smart Decisions. Scrum and XP alone aren't enough!"



We need a **framework** to help us make Smart Decisions

- **Measuring Progress towards Goals** - Defining measurable goals and recording before and after metrics to see if our solution really delivered value
- **Judicious with our Budget** - With our resources and investments of time and money to ensure they're focused on the right projects. We're not funding projects that can't quantify how their solutions produce measurable progress towards the prioritized business goals (If you can't deliver results with 10% of the budget, what makes you think you can deliver results with 100%?)
- **Analyze Frequent Feedback and Adapt** – Ensuring our investments are delivering measurable results using performance-to-cost ratios and percent-to-goals metrics. We're adapting to changing conditions on the ground using iterative planning and PDSA (Plan-Do-Study-Act)
- Utilizing **People, Process and Technology** – Using the right balance of each to deliver well thought out solutions that maximize overall operational performance and don't simply "speed up the mess"
- **Delivering value iteratively** - Utilizing popular agile methods (like Scrum and XP) to deliver the business value incrementally.

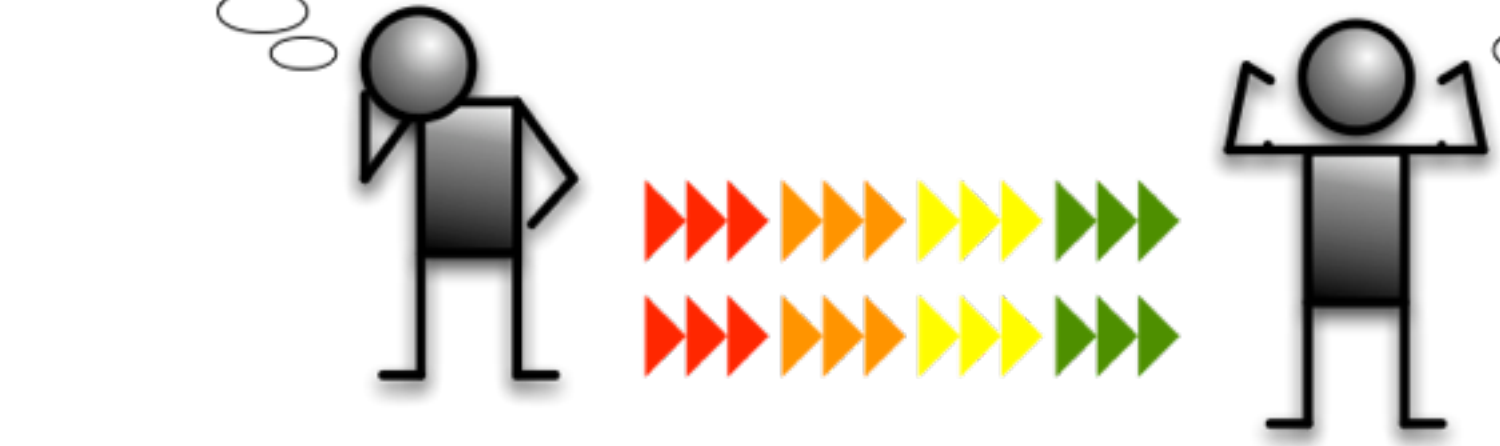
But first we need to think differently

Instead of thinking...

We want to do a project for a new system. What features will it have, how much is it going to cost and when will it be done?

We're thinking...

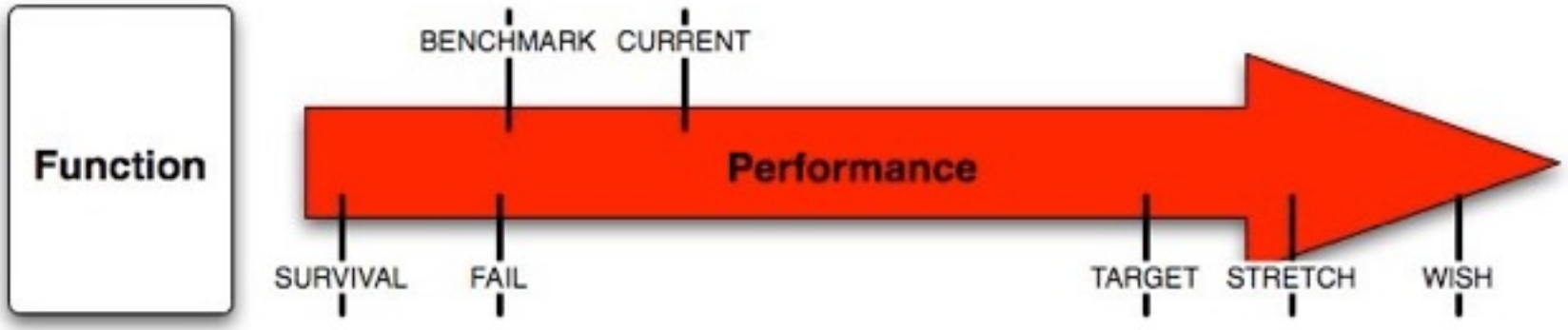
We've got a specific amount of money to invest in new projects. What are our highest priority business goals and where should we focus our resources to help us best achieve them?



Ryan Shriver

We deliver value using time-boxed iterations; and continue to fund projects *only if they deliver measurable business results each release*. Otherwise we *cancel the project* (and *preserve* our *resources* for another project)!

3 Requirements Examples DD Case: Specification with Planguage



Decisioning Capability:
Ambition: *Develop the capability to rapidly build and deploy new decisioning rules*
Scale: *Elapsed time in hours from idea to production upgrade of new decisioning rules that follow a pre-defined pattern*
Goal [End Project] : *< 1 hour*
Fail: *> 6 hours*
Meter: *Wall clock time*

Client Acquisition:
Ambition: *Acquire 2 new B2B clients and launch them on Release 2 of <Solution Name>*
Scale: *New clients put into production with transactions flowing between parties*
Goal [2008]: *2*
Fail [2008]: *0*
Meter: *Cognos report from analysis database*

Update Capability:
Ambition: *Ability for a trained business analyst to update the offer decision rules directly*
Scale: *Time in minutes for trained analyst to update offer rules and run test to validate change*
Goal [End 2008] 5 minutes
Fail: *> 15 mins*
Meter: *Elapsed time as measured from user interface using wristwatch*

Impact Estimation DD Case numeric evaluation of design

Goal: Increase Time to Sell (Individual hours devoted to direct sales activities) from 12 hrs/wk to 28 hrs/wk (30% to 70% of their time)		Design: Build New Accounts Wizard	Design: Electronically send data to SOR	Design	Totals
Current Benchmark	12 hrs / wk	12	12	12	
Target Goal	28 hrs / wk	28	28	28	
Scale Impact	hrs / wk	1	2.5	0	3.5
Scale Uncertainty	+ / - hrs/wk	0.5	1	0	1.5
Percentage Impact	on design	6%	16%	0%	22%
Percentage Uncertainty	percentage	3%	6%	0%	9%
Evidence	based upon	Anecdotal	High level estimate		
Source	person or doc	Ryan [06/18/07]	Ryan [06/20/07]		
Credibility	and 1	0.7	0.5		
Costs					
Solution Owner	effort hours	20	30	0	
Analysis	effort hours	70	200	0	270
Development	effort hours	100	300	0	400
Testing	effort hours	20	60	0	80
Total Resources	effort hours	210	590	0	800
Performance to Cost Ratio	of design	0.030	0.026	#VALUE!	
Credibility-adjusted Performance to Cost Ratio	factored in	0.021	0.013	#VALUE!	

Design Ideas

Impact Estimation

We need more than ‘Agile’, we need ‘Evolutionary’

Evolutionary Delivery

Evo Concepts

- Budgets
- Goals
- Design Ideas
- Impact Estimation
- Requirements Specification w/ Planguage
- Estimation Planning, & Tracking Estimation

Scrum Concepts

- Team Roles
- Stand Up Meetings
- Estimation Planning, & Tracking Estimation

Software Development Track

- *Extreme Programming (XP) Principles*
- *Test Driven Development (TDD) Principles*

Process Improvement Track

- *Lean Principles*

Scrum Team

Product Owner: Sets priorities
Scrum Master: Removes Blocks & Manage Process
Team: Develop business capability
Stakeholders: Observe & Advise

Daily Scrum
Every 24 Hrs

Daily Scrum(15 min):
What you did yesterday?
What you will do today?
Identify obstacles to work?

Sprint Backlog
List of task owned by Scrum Team
Only they can modify it
Built from Product Backlog

Sprint
1 to 4 Week
Duration

Sprint Backlog

Product Backlog

Chosen Design Idea

New Capability Demonstrated at the End of Each Sprint

Product Owner & Stakeholders

Task	Task	Task	Task	Task
Task	Task	Task	Task	Task
Task	Task	Task	Task	Task
Task	Task	Task	Task	Task

Program Backlog:
- Monitoring "Percent to Goals" and Budget
- Feedback into Phase 1 (Design Ideas)

"Plan and Deliver" *with Evolutionary Delivery*

Management Engineering

Plan *using Evo*

Define Success

Select Best Opportunity

Deliver *using Scrum*

Requirements Engineering

Design Engineering

Test Engineering

R

Release

Define Success

Consultancy Service Offerings

Project Inception

Lean Process Improvement

R

Agile Jump Start

Software Development

R

R

Measure Progress > Refine Objectives > Adapt Approach

4 8 12 16 20 24 28 32+

Time (weeks)

Evolutionary Delivery Components

Define Success

- Stakeholders
- Values
- Key Objectives
- Resources

Select Best Opportunity

- Design Ideas
- Design Criteria
- Impact Estimation
- Bang for the Buck

Deliver Value

Requirements Engineering

- Requirements Specification using Planguage
- Inspection
- Modeling
- All Qualities are Quantified

Scrum

- Release Planning
- Sprint Planning
- Sprints
- Stand Up Meetings
- Product Owner
- Retrospectives
- ScrumMaster
- Product Backlog
- Scrum Team
- Sprint Backlog

Design Engineering

- XP Principles & Practices
- Systems Architecture
- Test Driven Development
- Lean Principles & Practices

Management Engineering

- Evolve Goals
- Plan, Estimate, Track to Goals

Test Engineering

- Delivery Quality Measurement
- Automation

Value Driven Planning:

10 Value Principles

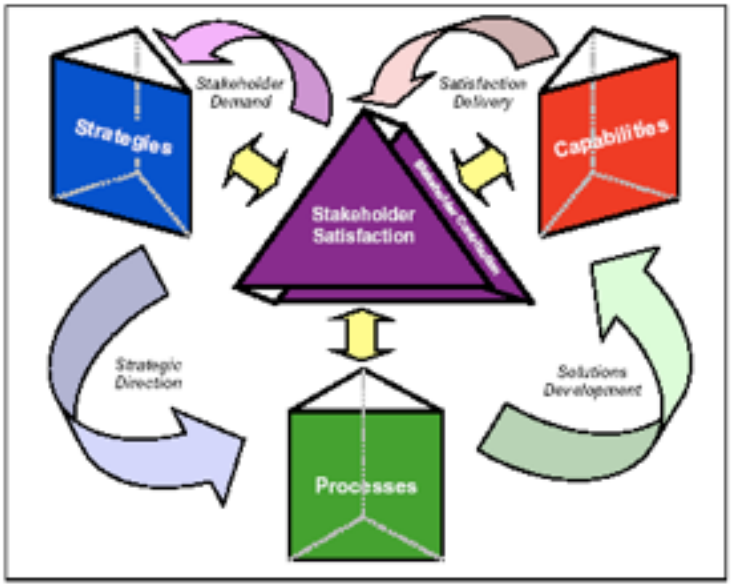
- Value Driven Planning focuses on
  - the primary values of key *stakeholders*.
- The *technology* used, and the project *processes* used are sub-ordinate.
- The critical stakeholder values are *quantified* and *trackable*.
- There is an assumption of
  - step by step achievement,
  - of *learning* at each step
  - and consequent *action*
    - to resolve problems of value achievement.

Gilb's 'Value Driven Planning' Principles:

1. **Critical Stakeholders determine the values**
2. **Values can and must be quantified**
3. **Values are supported by Value Architecture**
4. **Value levels are determined by timing, architecture effect, and resources**
5. **Value levels can differ for different scopes (where, who)**
6. **Value can be delivered early**
7. **Value can be locked in incrementally**
8. **New Values can be discovered (external news, experience)**
9. **Values can be evaluated as a function of architecture (Impact Estimation)**
10. **Value delivery will attract resources.**

Value Driven Planning Principles in Detail:

1. Critical Stakeholders determine the values

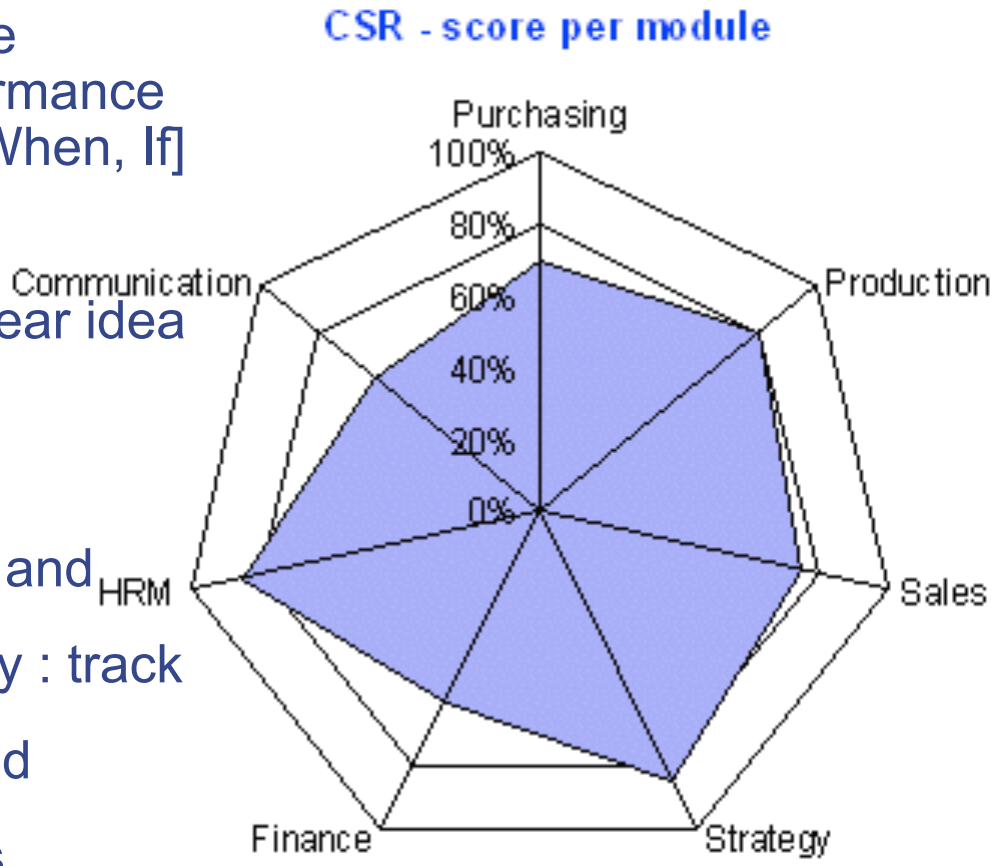


Critical: "having a decisive or crucial importance in the success or failure of something" <-Dictionary

- The primary and prioritized values we need to deliver are determined by
  - analysis of the needs and values of stakeholders
    - stakeholders who can determine whether we *succeed* or *fail*.
- We cannot afford to satisfy *other (less critical)* levels, at other times and places, yet.
  - Because that might undermine our ability to satisfy the more critical stakeholders –
  - and consequently threaten our overall project success.

2. 'Values' can and must be ***quantified***

- Values can, if you want, be expressed numerically.
  - With a defined scale of measure
  - with a deliverable level of performance
  - and with qualifier info [Where, When, If]
- Quantification is useful:
  - to clarify your own thoughts
  - to get real agreement to one clear idea
  - to allow for varied targets and constraints
  - to allow direct comparison with benchmarks
  - to put in Request for bids, bids and contracts
  - to manage project evolutionarily : track progress
  - as a basis for measurement and testing
  - to enable research on methods



- **Figure 1: Real (NON-CONFIDENTIAL version) example of an initial draft of setting the objectives that engineering processes must meet.**

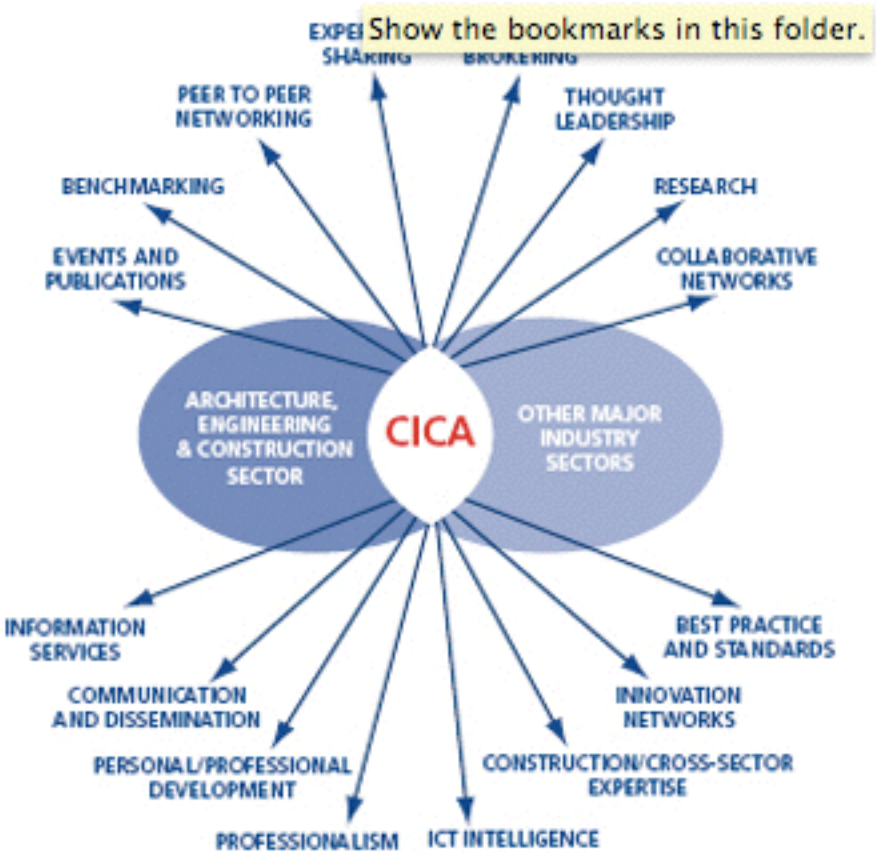
Business objective	Measure	Goal (200X)	Stretch goal ('0X)	Volume	Value	Profit	Cash
Time to market	Normal project time from GT to GT5	<9 mo.	<6 mo.	X		X	X
Mid-range	Min BoM for The Corp phone	<$90	<$30	X		X	X
Platformisation Technology	# of Technology 66 Lic. shipping > 3M/yr	4	6	X		X	X
Interface	Interface units	>11M	>13M	X		X	X
Operator preference	Top-3 operators issue RFQ spec The Corp	1	2	X		X	X
Productivity							X
Get Torden	Lyn goes for Technology 66 in Sep-04	Yes		X		X	X
Fragmentation	Share of components modified	<10%	<5%		X	X	X
Commoditisation	Switching cost for a UI to another System	>1yr	>2yrs				
Duplication	The Corp share of 'in scope' code in best-selling device	>90%	>95%		X	X	X
Competitiveness	Major feature comparison with MX	Same	Better	X		X	X
User experience	Key use cases superior vs. competition	5	10	X	X	X	X
Downstream cost saving	Project ROI for Licensees	>33%	>66%	X	X	X	X
Platformisation IFace	Number of shipping Lic.	33	55	X		X	X
Japan	Share of of XXXX sales	>50%	>60%	X		X	X
	Numbers are intentionally changed from real ones						

Business Values Quantified

3. Values are supported by Value Architecture

- Value Architecture: defined as:
  - anything you *implement* with a viev to satisfying stakeholder values.

- Value Architecture:
  - includes product/system objectives
    - Which are a 'design' for satisfying stakeholder values
  - Has a multitude of performance and cost impacts
  - can impact a given system differently, depending on what is in the system, or what gets put in later
  - Needs to try to maximize value delivered for resources used.

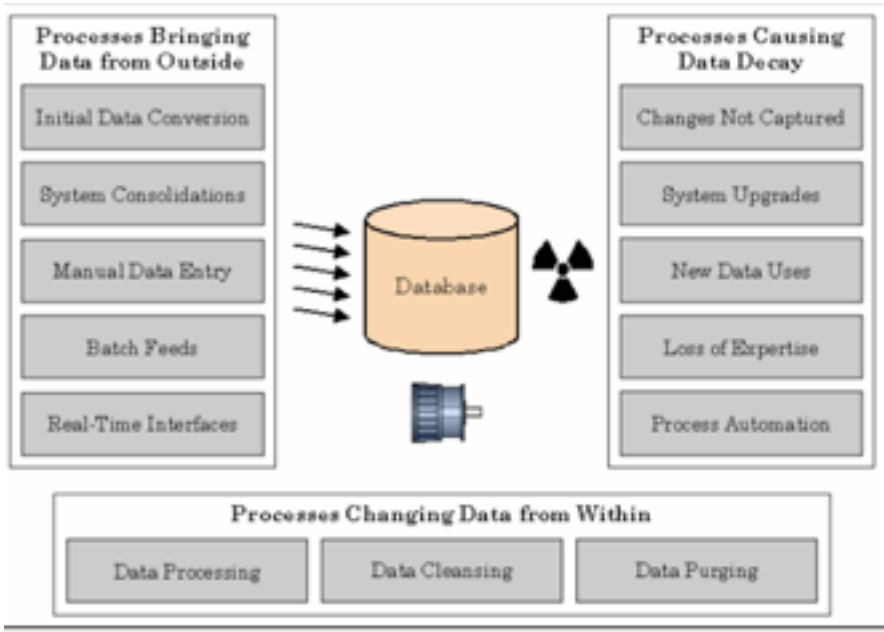


Value levels: defined as:

the degree of satisfaction of value needs.

Value level:

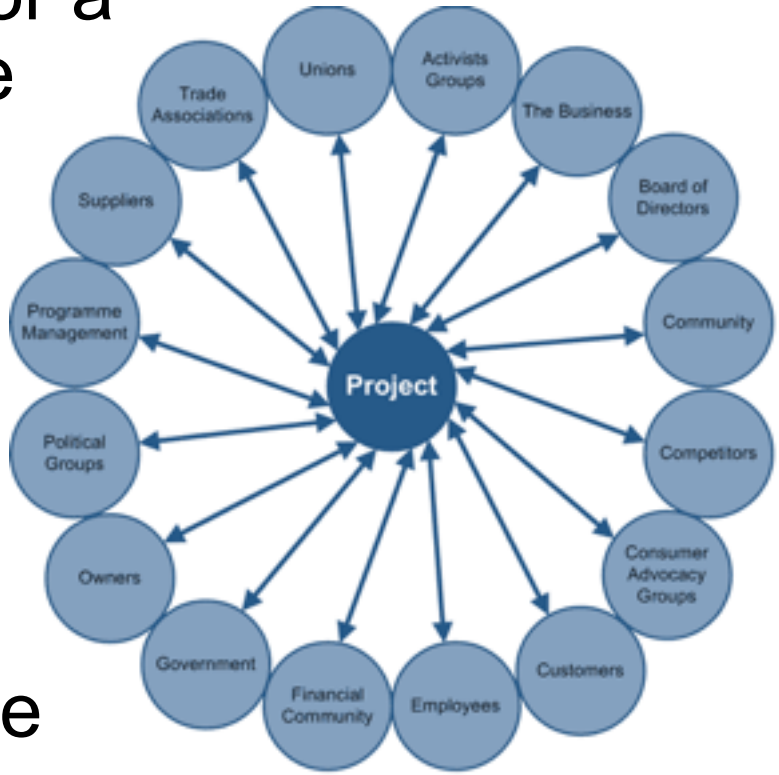
- depends on *when* you observe the level
  - The environment, the people, other system performance characteristics (security, speed, usability)
- depends on the *current incremental power* of *particular value architecture* components
- depends on *resources available* both in development and operation



The level of value needed, and the level of value delivered - for a single attribute dimension (like Ease of Use) can vary for:

- different stakeholders
- at different times
  - (peak, holiday, slack, emergency, early implementation)
- for different 'locations'
  - countries, companies, industries

There is nothing simple like 'one level for all'



- 6. Value can be delivered *early*

You do not have to wait until 'the project is done' to deliver useful stakeholder value satisfaction.

You can intentionally target the highest priority stakeholders, and their highest priority value area, and levels.

You can deliver them early and continuously

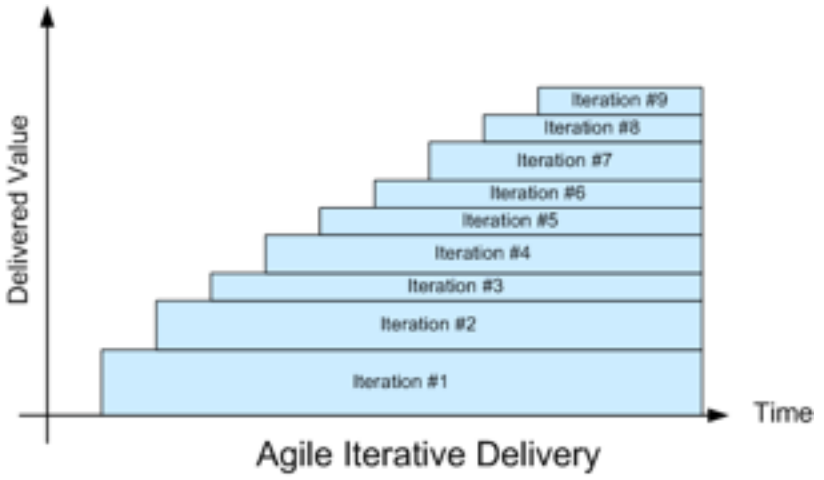
You can learn what is possible

And what stakeholders really value.

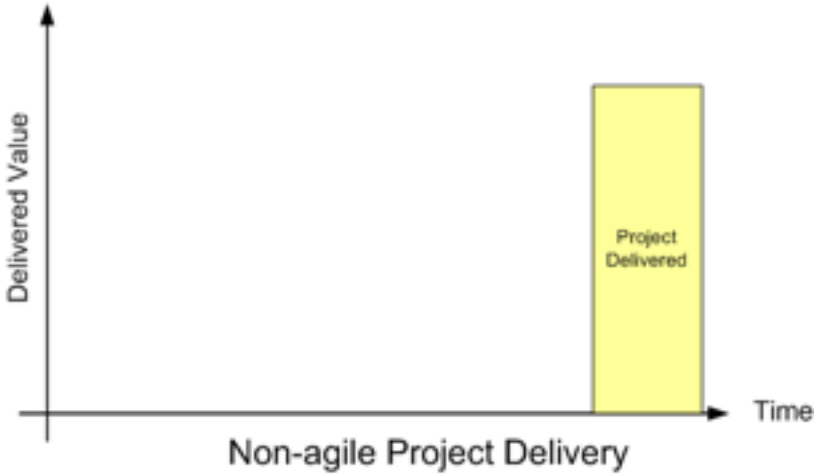
Discover new value ideas

Discover new stakeholders

Discover new levels of satisfaction



Agile Iterative Delivery

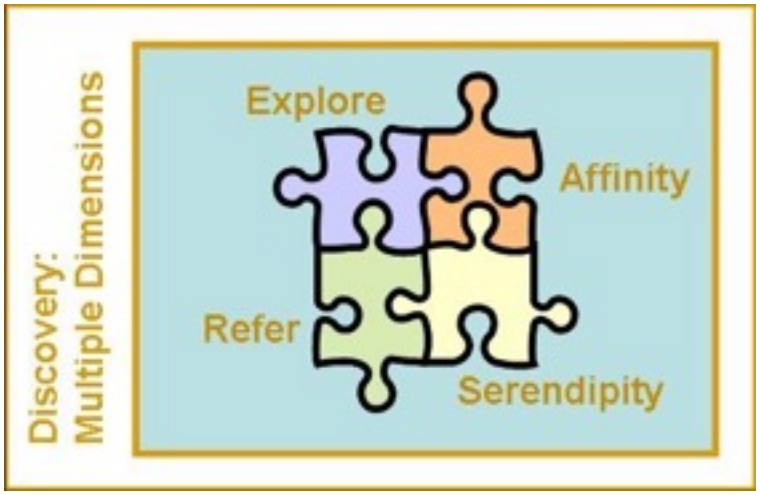


Non-agile Project Delivery

7. Value can be locked in incrementally

- You can increment the value satisfaction
  - –*towards* longer term Goal levels
- You can spread the value deliveries
  - – that are *proven* in *some* places,
  - –more widely in the next increments
- This probably assumes that you have really handed over real results to real people.
  - –Not just developed systems without delivery

- *Expect*, and try to discover,
  - –entirely new stakeholder values.
- These will of course emerge *after you start delivering* some satisfaction, because:
  - – Stakeholders believe you can help
  - –Things *change*



9. Values can be *evaluated* as a function of *architecture* (using 'Impact Estimation')

- It is possible to get an **overview** of
  - –the totality of impacts
  - – that your **architecture**
  - –(all designs and strategies)
  - – **might** have
  - – on all your defined stakeholder **n**
- Use an Impact Estimation table
  - – and you will be able to spot *opportunities* for
    - high value and
    - low cost early deliveries
      - – by analyzing the numbers on the table

		Viking Deliverables											
Business Objective	Weight	hardware adaptation	Telephony	Reference designs	IFace	Modularity	Defend vs Technology 66	Tools	User Experce	GUI & Graphics	Security	Defend vs OCD	Enterprise
Time to market	20%	20%	10%	30%	5%	10%	5%	15%	0%	0%	0%	5%	5%
Mid-range	10%	15%	0%	15%	0%	30%	15%	5%	10%	5%	5%	0%	0%
Platformisation Technology	5%	25%	10%	30%	0%	0%	10%	0%	5%	0%	10%	0%	5%
Interface	5%	5%	15%	15%	0%	5%	0%	5%	0%	0%	10%	0%	10%
Operator preference	10%	0%	10%	0%	15%	5%	20%	5%	10%	10%	20%	5%	10%
Get Torden	10%	25%	10%	10%	-10%	0%	20%	0%	10%	-20%	10%	10%	5%
Commoditisation	5%	20%	10%	20%	10%	-20%	25%	15%	0%	0%	5%	10%	5%
Duplication	10%	15%	10%	10%	0%	0%	40%	0%	0%	0%	5%	20%	5%
Competitiveness	5%	10%	15%	20%	0%	10%	20%	10%	10%	20%	10%	10%	10%
User experience	5%	5%	0%	0%	0%	20%	0%	0%	30%	10%	0%	0%	0%
Downstream cost saving	5%	15%	5%	20%	0%	10%	20%	0%	10%	0%	0%	10%	5%
Platformisation IFace	5%	10%	10%	20%	40%	0%	20%	5%	0%	0%	0%	0%	5%
Japan	5%	10%	5%	20%	0%	10%	0%	0%	10%	5%	0%	0%	0%
Contribution to overall result		15%	9%	17%	4%	7%	15%	6%	6%	1%	6%	6%	5%
Cost (£M)		£ 2.85	£ 0.49	£ 3.21	£ 2.54	£ 1.92	£ 2.31	£ 0.81	£ 1.21	£ 2.68	£ 0.79	£ 0.62	£ 0.60
ROI Index (100=average)		106	358	109	33	78	137	148	107	10	152	202	174

See next slide
For enlargement

Strategy Impact Estimation:
for a $100,000,000 Organizational Improvement Investment

Technical Strategies

Objectives

⇩ Defined In earlier slide

Viking Deliverables

Business Objective	hardware adaptation	Telephony	Reference designs	IFace	Modularity	Defend vs Technology 66	Tools	User Experience	GUI & Graphics	Security	Defend vs OCD	Enterprise
Time to market	20%	10%	30%	5%	10%	5%	15%	0%	0%	0%	5%	5%
Mid-range	15%	0%	15%	0%	30%	15%	5%	10%	5%	5%	0%	0%
Platformisation Technology	25%	0%	0%	0%	0%	10%	0%	5%	0%	10%	0%	5%
Interface	5%	15%	15%	0%	5%	0%	5%	0%	0%	10%	0%	10%
Operator preference	0%	0%	0%	15%	0%	20%	5%	10%	10%	20%	5%	10%
Get Torden	25%	10%	10%	-10%	0%	20%	0%	10%	-20%	10%	10%	5%
Commoditisation	20%	10%	20%	10%	-20%	25%	15%	0%	0%	5%	10%	5%
Duplication	15%	0%	10%	0%	0%	40%	0%	0%	0%	5%	20%	5%
Competitiveness	10%	15%	20%	0%	10%	20%	10%	10%	20%	10%	10%	10%
User experience	5%	0%	0%	0%	20%	0%	0%	30%	10%	0%	0%	0%
Downstream cost saving	15%	0%	0%	0%	0%	20%	0%	10%	0%	0%	10%	5%
Platformisation IFace	10%	10%	20%	40%	0%	20%	5%	0%	0%	0%	0%	5%
Japan	10%	5%	20%	0%	10%	0%	0%	10%	5%	0%	0%	0%
Contribution to overall result	15%	9%	17%	4%	7%	15%	6%	6%	1%	6%	6%	5%
Cost (£M)	£ 2.85	£ 0.49	£ 3.21	£ 2.54	£ 1.92	£ 2.31	£ 0.81	£ 1.21	£ 2.68	£ 0.79	£ 0.62	£ 0.60
ROI Index (100=average)	106	358	109	33	78	137	148	107	10	152	202	174

"Benefits"

Strategy Impacts on Objectives

Cost

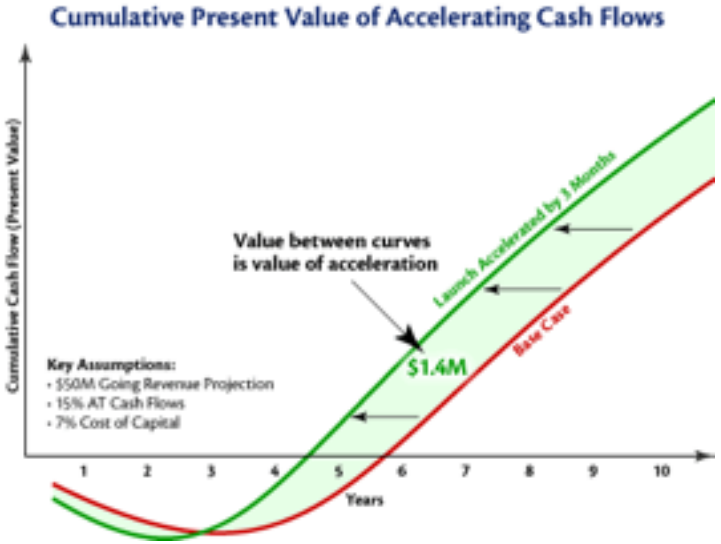
358 !

10. Value delivery will attract resources.

- If you are really good at delivering value
  - –You can expect to attract
    - even more funding
  - –Managers like
    - to be credited with success
  - – Money seeks
    - best interest rates

Gilb's Value Manifesto: A Management Policy?

1. Really useful value, for real stakeholders will be defined measurably.
   No nice-sounding emotive words please.
2. Value will be seen in light of total long term costs
   as a decent return on investment.
3. Powerful management devices, like motivation and follow-up, will make sure that the value for money is really delivered –
   or that the failure is punished, and the success is rewarded.
4. The value will be delivered evolutionarily –
   not all at the end.
5. That is, we will create a stream of prioritized value delivery to stakeholders, at the *beginning* of our value delivery projects;
   and continue as long as the real return on investment is suitably large.
6. The CEO is primarily responsible for making all this happen effectively.
   1. The CFO will be charged with tracking all value to cost progress.
   2. The CTO and CIO will be charged with formulating all their efforts in terms of measurable value for resources.



Source "Value Delivery in Systems Engineering" available at www.gilb.com
Unpublished paper http://www.gilb.com/tiki-download_file.php?fileId=137

The Value Delivery Problem

- Sponsors who order and pay for systems engineering projects, must justify their money spent based on the expected consequential effects (hereafter called ‘value’) of the systems.
-
- The value of the technical system is often expressed in presentation slides and requirements documents as a set of nice-sounding words, under various titles such as “System Objectives”, and “Business Problem Definition”

Some Assertions

Assertion 1. When top management allows large projects to proceed, with such badly formulated primary objectives, then

- they are responsible as managers for the outcome (failure).
- They cannot plead ignorance.

Assertion 2. The failure of technical staff (project management) to react to the lack of primary objective formulation by top management is also a total failure to do reasonable systems engineering.

- Management might have a poor requirements culture, but we should routinely save them from themselves.

Assertion 3. Both top managers and project personnel can be trained and motivated to clarify and quantify critical objectives routinely.

- But until the poor external culture of education and practice changes, it may take strong CEO action to make this happen in your corporation.
- My experience is that no one else will fight for this.

Assertion 4. All top level system performance improvements, are by definition, variables.

- So, we can expect to define them quantitatively.
- We can also expect to be able to measure or test the current level of performance.
- Words like ‘enhanced’, ‘reduced’, ‘improved’ are not serious systems engineering requirements terms.

For example:
(Real, engineering system, but doctored for anonymity)

1. *Central to The Corporations business strategy is to be the world's **premier** integrated <domain> service **provider**.*

2. *Will provide a much more efficient **user** experience*

3. *Dramatically scale back the **time** frequently needed after the last data is acquired to time align, depth correct, splice, merge, recompute and/or do whatever else is needed to **generate** the desired **products***

4. *Make the system much **easier** to **understand** and **use** than has been the case for previous system.*

5. *A primary goal is to provide a much more **productive** system **development** environment than was previously the case.*

6. *Will provide a richer set of functionality for **supporting** next-generation logging **tools** and applications.*

7. ***Robustness** is an essential system requirement (see rewrite in example below)*

8. *Major improvements in **data quality** over current practices*

For Example:

I rewrote the top level system requirement in the above example using Planguage [Gilb 2005]:

"7. Robustness is an essential system requirement."

to be:

Rock Solid Robustness:

- **Type**: *Complex* Product Quality Requirement.
- **Includes**: {Software Downtime, Restore Speed, Testability, Fault Prevention Capability, Fault Isolation Capability, Fault Analysis Capability, Hardware Debugging Capability}.
-

Software Downtime:

Type: Software Quality Requirement. **Version**: 25 October 2007.
Part of: Rock Solid Robustness.
Ambition: to have minimal downtime due to software failures <- HFA 6.1
Issue: does this not imply that there is a system wide downtime requirement?

Scale: <mean time between forced restarts for defined [Activity], for a defined [Intensity].>

Fail [Any Release or Evo Step, Activity = Recompute, Intensity = Peak Level] 14 days <- HFA 6.1.1

Goal [By 2008?, Activity = Data Acquisition, Intensity = Lowest level] : 300 days ??
Stretch: 600 days.

Restore Speed:

Type: Software Quality Requirement. **Version**: 25 October 2007.
Part of: Rock Solid Robustness
Ambition: Should an error occur (or the user otherwise desire to do so), the system shall be able to restore the system to a previously saved state in less than 10 minutes. <-6.1.2 HFA.

Scale: Duration from Initiation of Restore to Complete and verified state of a defined [Previous: Default = Immediately Previous]] saved state.

Initiation: defined as {Operator Initiation, System Initiation, ?}. Default = Any.

Goal [Initial and all subsequent released and Evo steps] 1 minute?
Fail [Initial and all subsequent released and Evo steps] 10 minutes. <- 6.1.2 HFA
Catastrophe: 100 minutes.

Testability:

Type: Software Quality Requirement.
Part of: Rock Solid Robustness
Initial Version: 20 Oct 2006
Version: 25 October 2007.
Status: Demo draft,
Stakeholder: {Operator, Tester}.
Ambition: Rapid-duration automatic testing of <critical complex tests>, with extreme operator setup and initiation.

Scale: the duration of a defined [Volume] of testing, or a defined [Type], by a defined [Skill Level] of system operator, under defined [Operating Conditions].

Goal [All Customer Use, Volume = 1,000,000 data items, Type = WireXXXX Vs DXX, Skill = First Time Novice, Operating Conditions = Field, {Sea Or Desert}. <10 mins.

Design Hypothesis: *Tool Simulators, Reverse Cracking Tool, Generation of simulated telemetry frames entirely in software, Application specific sophistication, for drilling – recorded mode simulation by playing back the dump file, Application test harness console <-6.2.1 HFA*

the problem with conventional requirements

- their source or authority
  - may be *undocumented* and *unknown*
- they are probably not at all clear
  - about *exactly* what should happen,
  - where or when, or under which conditions
- there is no contract,
  - to pay *only* upon such *results* being *delivered*
- there is no specific design or architecture,
  - to *enable* the technical product to achieve the requirements

£50 million Wasted

- The above example was the basis in 1999 for a project that had
  - in 2006 spent over $100 million,
  - for 8 years
  - and had never delivered any value whatsoever to the corporation.
- There was never any quantified or testable definition of the requirements.
- There was never any direct link
  - from the project activity, requirements, or architecture,
  - to these primary top management
    - (CEO and next level directors) objectives.
- The project was doomed from the start.

Another Real (Doctored) Example: Financial Corp. Top Level Project requirements

1. *Reduce the costs associated with managing redundant / regionally disparate systems.*
2. *Single global portfolio management system.*
3. *Reduce overall spending with a reduction in redundant initiatives.*
4. *Governance structures - system agnostic.*
5. *All projects in project portfolio system.*
6. *Reduce development project spend on low priority work with better alignment between Technology and business demand.*
7. *Project portfolio Framework, Business Value metrics for prioritization.*
8. *Reduction in cost over runs.*
9. *Definition criteria for project success.*
10. *Metrics and exception reporting for cost management.*
11. *Linkage of actual costs to forecast.*
12. *Increase revenue with a faster time to market.*
13. *Knowledge management, project ramp up templates.*

The Financial System

- This project spent about $50 million, in a single year.
- Responsible management, impatient for some results, discovered to their horror, through an audit, that the above primary objectives had **never been clarified or taken seriously.**
- The responsible ('former') project manager had chosen to **ignore** the opportunity, planned by a major component supplier, to **clarify** these objectives.
- The project manager spent a lot of effort obtaining 'requirements from users',
  - but no further effort on *these* **primary** objectives above.
- Serious effort was, after the audit, then immediately spent quantifying and taking seriously these primary objectives.
- It took a single day to draft a quantified version.
- The quantified version made a clear distinction between
  - technical objectives (system quality – examples 2 and 5 above) and
  - stakeholder values (making the business better, examples 8 and 12 above).

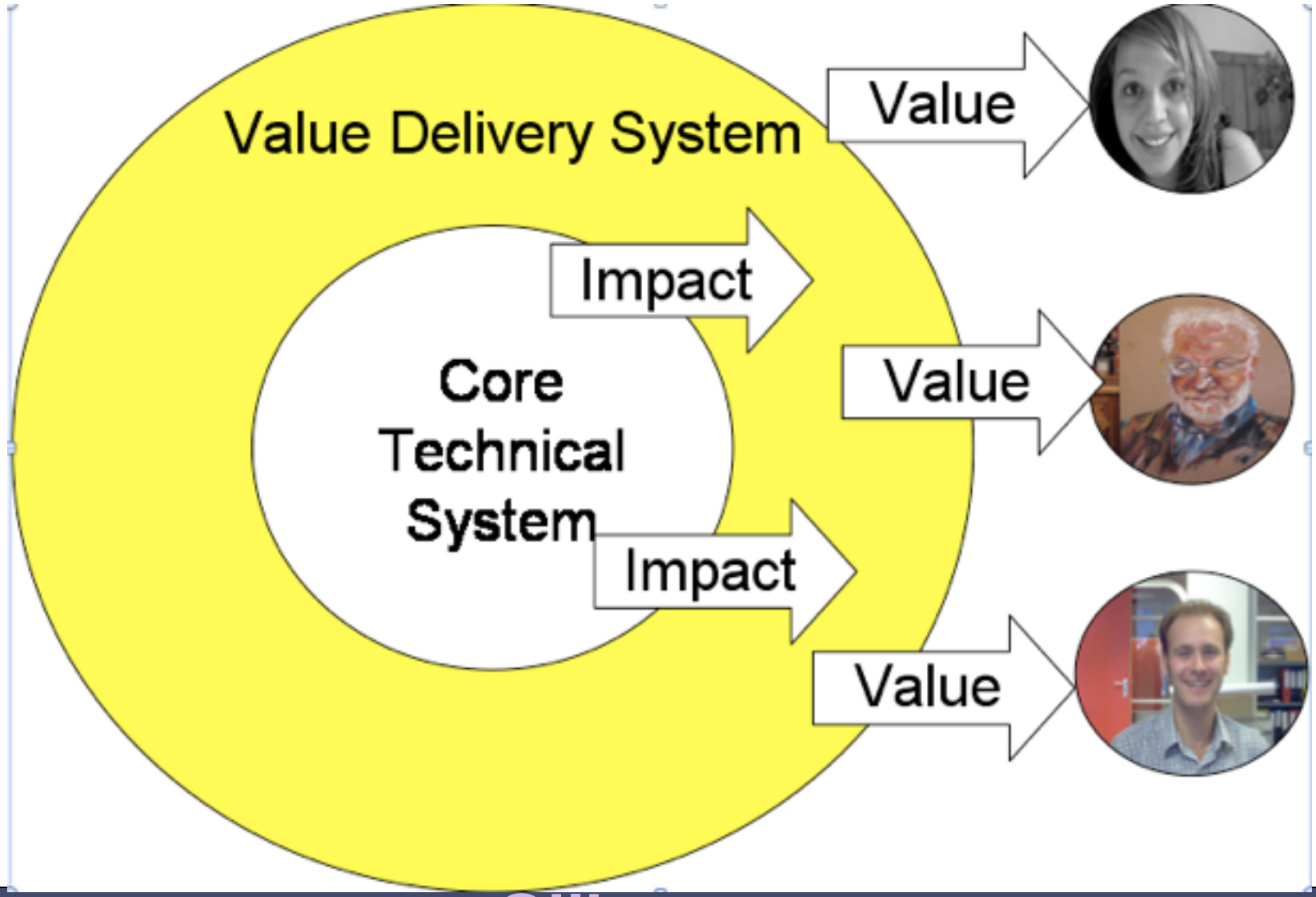
Another Assertion
Delivering Value

- Assertion 5.
  - If the hardware/software systems supplier is
    - not prepared to deal with the system level that delivers the value from their product,
    - then someone,
      - internally or an external contractor
    - needs to undertake the project of delivering the value expected.

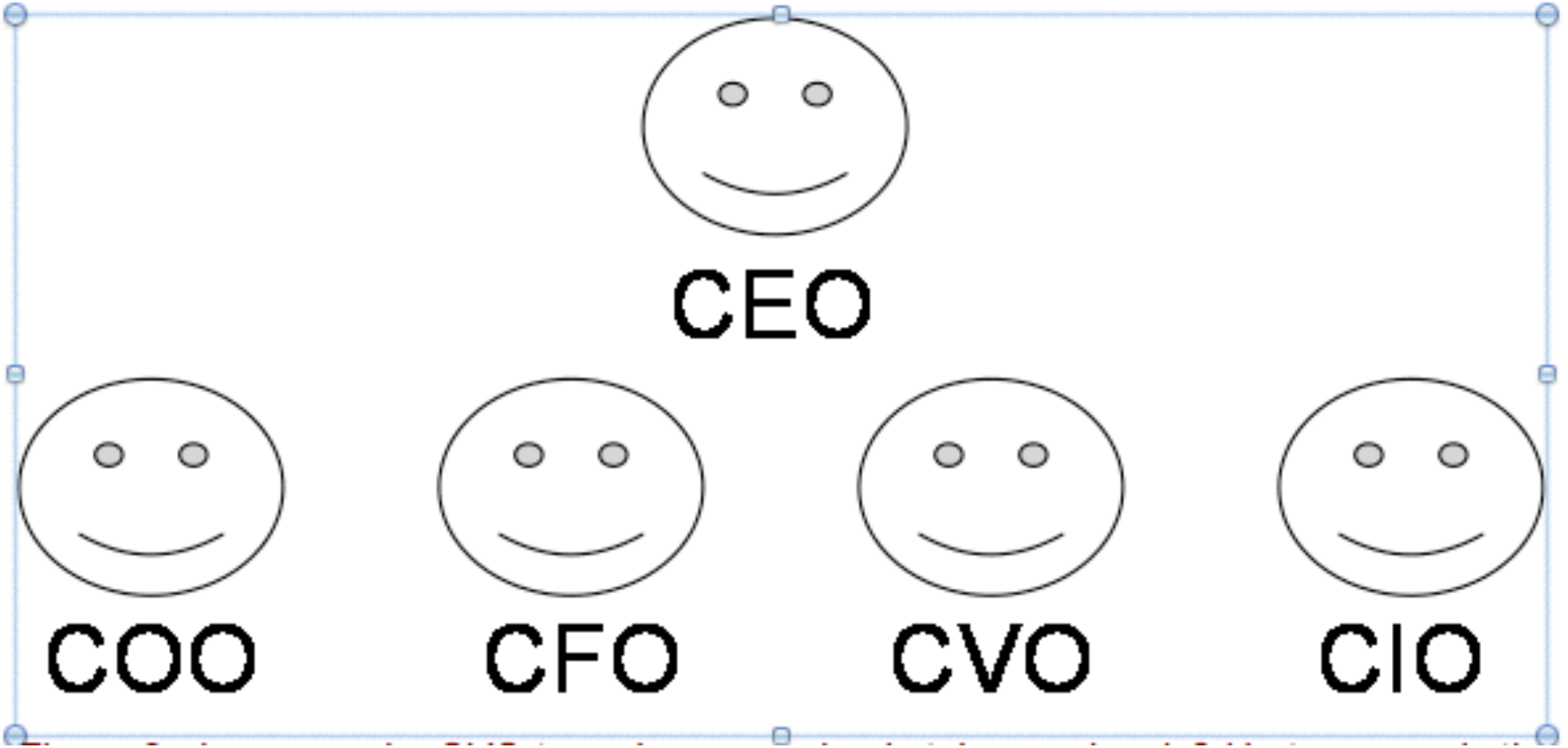
Assertion 6.
Systems Engineering for Value

- This ‘value delivery process’ is
  - likely to entail considerable human and organizational aspects,
  - and little hardware and software technology.
- So it may be inappropriate work for systems engineers
  - who are not expert in, and committed to, the social, political, and organizational aspects of systems engineering.
- But of course this ‘social’ ability
  - is a necessary and valid component of full systems engineering –
  - or we cannot call it ‘systems’ engineering
  - and exclude the social, political system aspects.
-

Value delivery is NOT Technical Construction



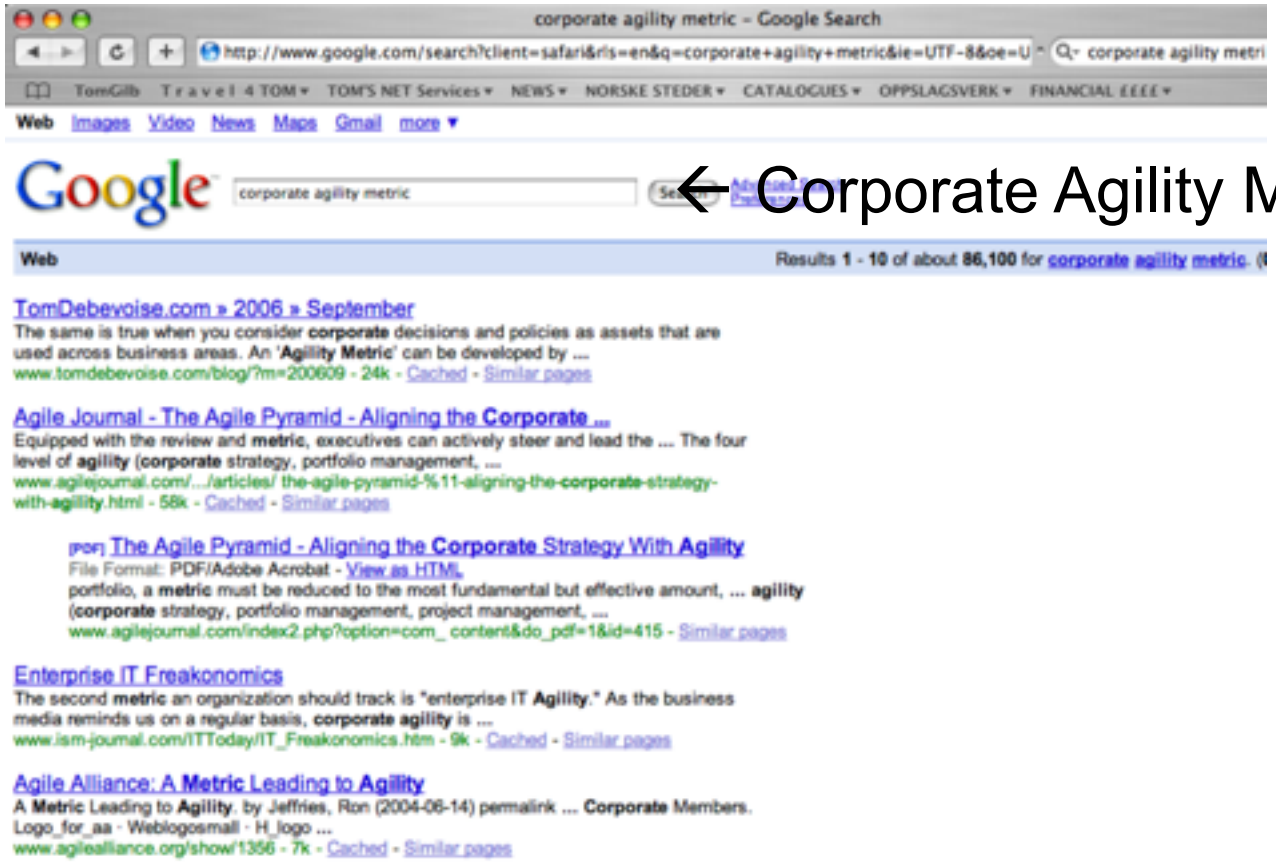
Do we need a Chief Value Officer?



The Value Principles:

1. Value can always be articulated quantitatively, so that we can understand it, agree to it, track it, contract for it and understand it in relation to costs.
2. Value is a result, delivered to a real set of stakeholders.
3. Value must be seen in light of lifetime total cost aspects, and must be as profitable as alternative investments.
4. Value occurs through time, as a stakeholder experience; it is not delivered when a system to enable it is delivered – only when that system is successfully used to extract the value.
5. Value can be delivered early, and for part of one stakeholder's domain. This proves the value potential, and actually improves the real organization.
6. There is never a really sufficient reason to put off value delivery until large-scale long-term investments are made. This is just a common excuse from the many weak, ignorant, cowards who would like to spend a lot of money before being held to account.
7. People who cannot deliver a little value early, in practice, cannot be entrusted to deliver a lot of value for a larger investment.
8. The top management must be primarily responsible for making value delivery happen in their organization. The specialist managers will never in practice take the responsibility, unless they are aiming to take over the top job.
9. Value is a multiplicity of improvements, and certainly not all related to money or savings – but we still need to quantify the value proposition in order to understand it, and manage it.
10. If we prioritize highest value for money first, then we should normally experience an immediate and continuous flow of dramatic results, that the entire organization can value and

1. Value can always be articulated quantitatively, so that we can understand it, agree to it, track it, contract for it and understand it in relation to costs.



← Corporate Agility Metric

- If all else fails, Google it!

2. Value is a result delivered to a real set of stakeholders.

- Value is not 'activated' by a technical performance characteristic alone,
  - like Usability, security or Robustness.
- It is only created when it meets real people in their everyday stakeholder situation of work:
  - Call Center, Battlefield Analyst, Corporate Trader.
- It has to save them time, or make their work better.
- The value created by the interaction with a stakeholder type may be cumulated every time the system is used for some new activity, customer, transaction, or decision.
- It may be cumulated by a very large number of that type of stakeholder (10,000 sales people). And through a very long time (years).
- It is obvious from this common sense observation that value is *not* created by the technical system performance characteristics (speedy response, user friendly),
  - but by making those technical system characteristics available
    - in practice
    - to as many real people, and
    - as many transactions, and
    - for as long a time as possible.

- We cannot allow ourselves to be blinded narrowly by quantified value.
- We must constantly estimate, and manage the value for money: the return on investment.
- And if the costs of delivering the value get out of hand, and exceed the value –
  - it is time to either reengineer the system
  - or decommission it.
  - Who will do this if not some constant CVO vigilance?

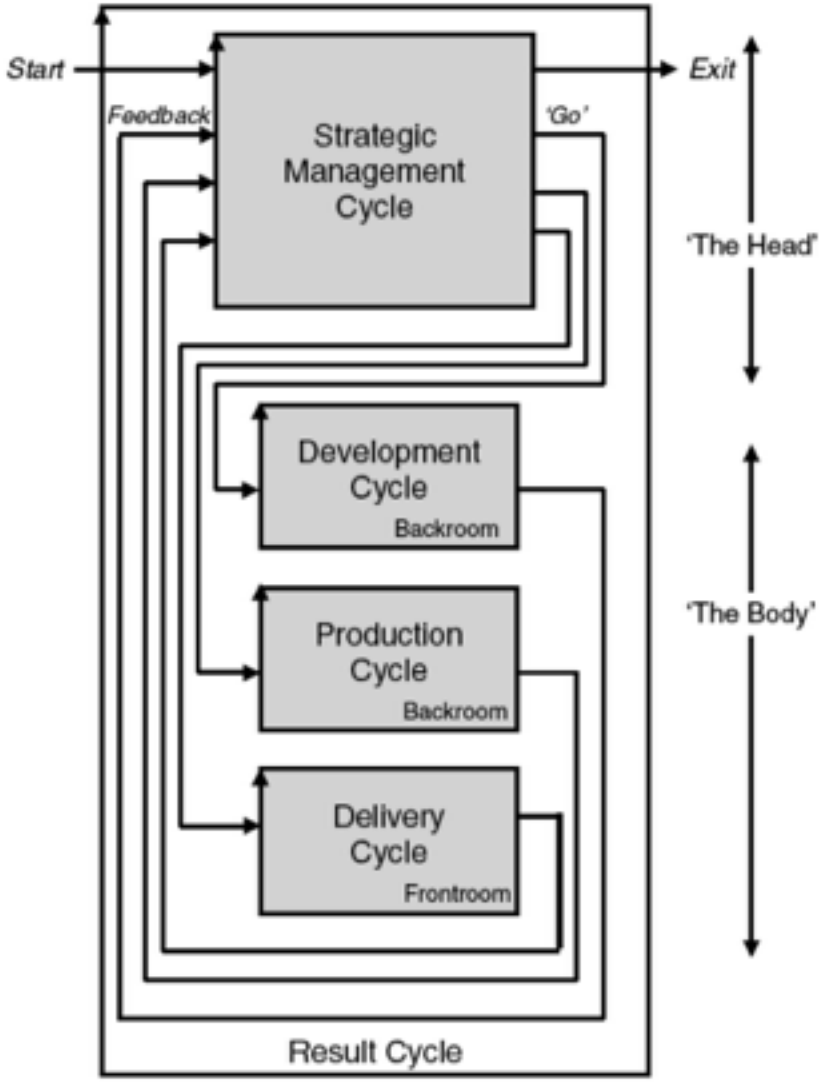
4. Value occurs through time, as a stakeholder experience: it is not delivered when 'a system to enable it' is delivered – only when that system is successfully *used* to extract the value.

- A **conscious strategy**, and **conscious formal plan**, must be made to deploy a technical system so that the value is delivered.
- We have to deal with political problems – like power centers (trade unions, management fiefdoms) and economic waste centers.
- We have to motivate people to give up their comfortable older systems and deploy scary new ones.
- We have to support the correct use by
  - training, call centers, local consultancy, measurement and feedback on the technical system,
  - is it actually delivering what we need, in order to get people to use it at all, to use it well?
- feedback on the stakeholder environments it is deployed in:
  - are they happy with it?
  - Do they have improvement suggestions?
  - Are there undesired variations in costs and benefits?
- feedback on deployment to the entire scope of stakeholders,
  - in relation to time plans:
  - is it being deployed successfully rapidly enough?
-
- Obviously this should be the natural concern and use of true systems engineering.
  - But in fact, there is little in the training, the conferences, the handbooks [INCOSE SE Handbook], to verify that systems engineering as a discipline has matured to the point where these concerns are safely included.
  - We are still too much 'engineers' (techies); and know and care too little about value management, and the organizational and management culture part of our domain.

5. Value can be delivered early, and for *part* of one stakeholder's domain. This proves the value potential, and actually improves the real organization.

- Our systems development culture is still very much a 'waterfall' culture.
- Finish the big system, and then deploy it [INCOSE SE Handbook 2-3, and 3-2 for example].
- There was no visible mention, in the Handbook, of a true evolutionary life cycle (even though the US DoD adopted one for software at least long ago, DoD Mil Std 498).
- There is no notion of early, frequent and gradual delivery of results to stakeholders, even though that has been practiced successfully in many large military, space and software systems for decades [Larman].
- Big Bang is still our mentality.
- I helped Douglas/Boeing to do value delivery Evolutionary projects for 25 aircraft projects in 1990. It was an unknown concept for them, but it was easily doable by every team we did it on; in real projects. We use 'next week' as our measure of when we would produce some useful value.
- I know that this sounds incredible and impossible to conventional ears. But it is simple enough in practice, and very close indeed to weaponry progress during the Second World War [Discovery Channel!].

Intelligent Feedback About Value



6. There is never a really sufficient reason to put off value delivery until large-scale long-term investments are made.
This is just a common excuse from those who would like to spend a lot of money before being held to account.

- There are vested interests who will happily consume public and private corporate money forever and deliver failure or little or no real value.
- The consumer and their representatives seem happy to contract for *effort*, but not contract for *value*.
- I cannot believe there are so many foolish people with so much money as I have had occasion to observe in practice
  - (example the $50 to $100 million wasted projects at the beginning of this paper, which are in fact small by comparision with some; like documented DoD waste in software engineering alone ($20 billion annually, many years ago).
- This is not necessary! We could avoid it by contracting for value and results. [Gilb, No Cure No Pay]. This is hardly on the agenda, and not discussed at all in the INCOSE Handbook.
- It would require two technical pieces of knowledge
  - The ability to quantify and measure value
  - The ability to decompose large projects into much smaller increments of value delivery.
- These exist, but the 'will to contract for value' does not.
- Some management leadership please!

- Ericsson of Sweden, who learned to deliver mobile telephone base stations in 1990 in monthly evolutionary steps observed this principle (Jack Järkvik).
- If you are going to spend $100,000,000 before anything happens, and nothing then does.
  - It might have been a good idea to offer the project or supplier a mere $1 million (1%)
    - and ask if they could create some of the long-term projected value for that 1% of budget.
    - If they cannot, then there is no reason to believe they will use your $100 million wisely.
    - If they can; do so, then feed them millions, one at a time until it is no longer profitable!

- Top management, the CEO, needs to decide they are primarily responsible for value for money, and dictate a policy of focus on 'value for money' (see earlier in this paper for policy ideas).
- One excellent CEO client of mine who did so, Robb Wilmott of ICL UK (23,000 employees then), turned years of losses into 14 straight years of profit for his computer company – unlike competitors, like IBM, at the time. My observation was:
- • it only happened because the CEO threatened all other top managers with loss of power and budget if they did *not* 'quantify the value' they were going to deliver
- • they began to think clearly about their responsibilities, perhaps for the first time
- • it helps if the CEO is an engineer, not an MBA ☺️
- Another UK CEO, pulled the same trick – about 2003.
  - But had to fire the marketing director, and the sales director, for refusing to really play ball.
  - Some directors have a real fear of being specific about what they are responsible for.
  - Interestingly the current Chairman of *this* company was one of the above-mentioned ICL Directors (Marketing) who we trained to quantify, things like the primary new product line vision, 'Adaptability' of his product.

9. ‘Value’ is a multiplicity of improvements, and certainly not all related to money or savings – but we still need to quantify the value proposition in order to understand it, and manage it.

- I strongly dislike value schemes that try to turn all values into money. Do they really think management understands no other concept?
-
- Peter Drucker, I think it was (Management By Objectives, in ‘The Practice of Management’), established long ago that no corporation is driven by money alone. Thus the Balanced Scorecard, to retain some non-financial balance, I suppose.
- If the value you are aiming at is for example, ‘increased potential customer willingness to shortlist you’,
  - then there is an estimable money value for that,
  - but I would be afraid of losing focus on the short-listing, by converting this idea to money.
- You would need to measure the quantity of real short-listing to manage that value, for example.
  - I believe you need to state and measure things directly,
  - especially of you want to track early lead indicators of value –
  - and keep people focused on a dynamic and changing situation.

10. If we prioritize highest value for money first, then we should normally experience an immediate and continuous flow of dramatic results, that the entire organization can value and relate to. Be deeply suspicious of long-term visions with no short-term proof.

- We should try to skim the cream off the top.
  - With early realistic feedback, and changing technology and markets, we should be able to avoid a dramatic diminishing return on investment for some time.
- Projects, at one extreme, should be practically self-funding;
  - or at least not in need of huge initial budgets, then overspent by factor 3.14 (Pie instead of 'piece of cake') before management feels uncomfortable
- You have a lot of choice, in spite of some dependencies,
  - to 'cherry pick' very high value for money, early deliveries.
  - Not exactly a new marketing technique –
    - but maybe alien to our Defence Supplier Systems Engineering mentality.
- Again, if we contracted to pay them for value for money,
  - they would be more focussed on making it happen.
  - This is *our* problem, not theirs.
  - We fail to motivate suppliers to do the right thing for us.
- We fail to even discuss this in our systems engineering literature.
  - We have progress payments, but not based on value delivery, early and frequently.
  - 'Payment Schedules' (sounds nice and bureaucratic) are mentioned in the SE Handbook, but not 'Value Payments'.
  - We need to extend the concept!

Summary

- Top management needs to change their culture
  - to manage the actual delivery of real value,
  - and not leave it to systems engineers to drive this change.
- Systems Engineers can execute the value engineering and delivery –
  - but only top management can make it happen.